A Corrida da Inteligência Artificial Saiu dos Modelos e Entrou nos Sistemas

E isso muda não apenas como programamos, mas quem controla o trabalho intelectual

Por Dennys Pavanelli

Artigo editorial - Junho de 2026

Tese central

A vantagem competitiva da IA moderna não está apenas no modelo. Está no sistema que conecta modelo, dados, interface, ferramentas, contexto, execução, segurança e fluxo real de trabalho.

A Corrida da Inteligência Artificial Saiu dos Modelos e Entrou nos Sistemas

E isso muda não apenas como programamos, mas quem controla o trabalho intelectual

Durante muito tempo, a disputa entre empresas de inteligência artificial foi narrada como uma competição simples: qual modelo responde melhor, qual gera código mais limpo, qual vence benchmarks, qual parece mais inteligente em uma conversa.

Essa narrativa ainda existe. Ela aparece em vídeos, manchetes, posts virais e comparações de ranking. Mas já não explica o centro real da transformação.

A nova corrida da inteligência artificial não é mais apenas entre modelos. É entre sistemas.

O que está em jogo agora não é somente quem tem o melhor chatbot, mas quem conseguirá construir o ambiente completo onde o trabalho humano será planejado, executado, revisado e automatizado. No caso da programação, isso significa sair da era dos copilotos, que sugerem trechos de código, e entrar na era dos agentes, que conseguem navegar por projetos inteiros, alterar arquivos, executar ferramentas, testar hipóteses, corrigir erros, coordenar subtarefas e manter contexto por longos períodos.

Essa mudança altera o papel do desenvolvedor, a estrutura das empresas de software, a economia da criação digital e a distribuição de poder entre grandes plataformas tecnológicas.

A disputa deixou de ser apenas sobre inteligência. Agora é sobre infraestrutura de trabalho.

Do snippet ao agente

A primeira fase da IA para programação foi dominada por ferramentas que completavam código. O usuário escrevia uma função, e a IA sugeria a continuação. Depois vieram os assistentes conversacionais, capazes de explicar erros, gerar scripts, refatorar trechos e ajudar na documentação.

Essa fase foi poderosa, mas ainda mantinha o humano como operador central de cada passo.

Agora a ambição é outra.

Os novos agentes de programação não querem apenas responder perguntas. Eles querem trabalhar dentro do projeto. Um agente moderno pode receber uma tarefa como “corrija o bug no fluxo de autenticação”, explorar os arquivos relevantes, identificar dependências, propor um plano, editar código, rodar testes, analisar falhas e repetir o ciclo até chegar a uma solução aceitável.

Isso não transforma automaticamente a IA em um engenheiro completo. Mas desloca o trabalho humano para outro nível.

Em vez de apenas escrever código linha por linha, o profissional passa a supervisionar, especificar, validar e orquestrar sistemas inteligentes.

O desenvolvedor deixa de ser apenas executor e se torna diretor técnico de agentes.

Mas essa frase, por mais elegante que pareça, ainda esconde uma tensão: quem define o limite entre direção humana e execução autônoma? Em que ponto o agente deixa de ser ferramenta e passa a influenciar decisões técnicas que o humano apenas aprova por cansaço, confiança excessiva ou falta de tempo?

Essa será uma das perguntas centrais da próxima fase.

O modelo ainda importa, mas já não basta

Durante anos, a indústria tratou escala como sinônimo de progresso. Mais parâmetros, mais dados, mais GPUs, mais tokens. A escala continua importante, mas já não é suficiente.

Um modelo gigantesco pode ser impressionante em respostas isoladas e ainda assim falhar em tarefas longas, cheias de dependências, ambiguidade e necessidade de correção contínua.

Na programação real, o problema raramente é escrever uma função perfeita em isolamento. O problema é entender um sistema existente. É saber onde mexer sem quebrar outra parte. É lidar com código legado, dependências mal documentadas, testes incompletos e decisões de arquitetura tomadas meses ou anos antes.

Por isso, o diferencial passa a estar nos dados de comportamento real e na integração profunda com o ambiente de trabalho.

Um modelo treinado apenas em repositórios públicos aprende o produto final: o código.

Um sistema que observa workflows reais aprende o processo: como desenvolvedores investigam bugs, quais arquivos abrem primeiro, como testam hipóteses, como descartam caminhos ruins, como estruturam mudanças e como revisam o próprio trabalho.

Essa é uma diferença enorme.

Código público mostra o resultado.

Workflow mostra o raciocínio operacional.

E raciocínio operacional é muito mais valioso do que código isolado.

O sistema é o produto

A movimentação de empresas como xAI, Alibaba, DeepSeek, OpenAI, Anthropic, Google e outras mostra que o centro da competição está mudando.

O modelo continua sendo importante, mas o produto real começa a ser o sistema onde o modelo age.

Um agente integrado ao terminal, ao repositório, aos testes, ao histórico do projeto, aos diffs, à documentação, aos servidores de ferramentas e aos fluxos internos de uma empresa é muito mais poderoso do que um chatbot isolado.

Ele não está apenas “respondendo sobre código”.

Ele está operando no local onde o trabalho acontece.

Isso transforma a disputa por ferramentas de desenvolvimento em uma disputa por infraestrutura estratégica. Quem controla a ferramenta que o programador usa diariamente controla também uma camada valiosa de sinais: padrões de correção, fluxos de refatoração, tipos de erro, hábitos de navegação em código, pontos de bloqueio e formas reais de resolver problemas.

Mas aqui é preciso manter rigor.

Há uma diferença entre dizer que dados de workflow podem gerar vantagem competitiva e afirmar, sem evidência completa, que todo comportamento sensível dos usuários está sendo usado para treinamento. A segunda afirmação exige transparência, consentimento explícito, análise jurídica e políticas claras.

O ponto central não é acusar uma empresa específica.

O ponto central é perceber a direção estrutural: a vantagem competitiva da IA não será apenas o modelo. Será a capacidade de conectar modelo, dados reais, ambiente de execução, ferramentas, contexto, memória, permissões, segurança e confiança.

O sistema é o produto.

A corrida pelos agentes de longa duração

A próxima fronteira não é apenas responder bem. É permanecer útil por muito tempo.

Agentes de longa duração são diferentes de modelos bons em respostas curtas. Eles precisam manter objetivo, lembrar decisões, evitar loops, recuperar-se de erros e continuar avançando depois de muitas etapas.

Essa é uma das fronteiras mais difíceis da IA atual.

Um modelo pode parecer brilhante em uma demonstração de cinco minutos e falhar em uma tarefa de cinco horas. Pode gerar uma interface bonita na primeira tentativa, mas se perder quando precisa depurar um sistema real. Pode acertar um benchmark e ainda ser pouco confiável em produção.

Por isso, a pergunta mais importante daqui para frente não é apenas “qual modelo venceu o ranking?”.

As perguntas realmente importantes são outras:

- Por quanto tempo ele consegue trabalhar de forma útil sem supervisão intensa?
- Quantas etapas executa sem entrar em loop?
- Quanto custa cada tarefa concluída?
- Qual é sua taxa de erro em projetos reais?
- Ele sabe parar quando está errado?
- Ele explica o que mudou?
- Ele deixa rollback possível?
- Ele registra decisões?
- Ele sabe pedir ajuda quando o contexto é insuficiente?
- Ele consegue distinguir incerteza real de confiança fabricada?

Essas perguntas são menos glamorosas do que um placar de benchmark. Mas são elas que determinarão a utilidade econômica real dos agentes.

Autonomia não é apenas continuar executando.

> **Autonomia útil é continuar executando sem destruir confiança, custo e controle.**

DeepSeek e o espelho da pesquisa feita por agentes

O caso dos papers e frameworks de agentes autônomos de pesquisa é simbólico porque mostra uma virada curiosa: uma IA ajudando a mapear o futuro das IAs que fazem pesquisa.

É uma metáfora quase perfeita do momento atual.

Mas o ponto mais importante não é concluir que a ciência humana acabou. Essa seria uma leitura apressada. O ponto correto é mais sutil: agentes já conseguem produzir artefatos acadêmicos estruturados, plausíveis e úteis como ponto de partida, mas ainda dependem de validação humana para rigor, originalidade, interpretação e responsabilidade.

> **Há uma diferença entre parecer ciência e ser ciência validada.**

Um agente pode organizar literatura, propor taxonomias, resumir campos inteiros e escrever com aparência profissional. Mas ciência exige mais do que forma. Exige método, validação, reprodutibilidade, crítica, comparação, revisão e responsabilidade intelectual.

Esse ponto é essencial porque a IA moderna é especialmente boa em produzir aparência de competência.

Ela escreve bem. Estrutura bem. Resume bem. Argumenta com fluidez.

Mas fluidez não é prova. Organização não é verdade. Segurança textual não é confiabilidade científica.

É justamente aí que aparecem os limites atuais dos agentes: loops cognitivos, dificuldade de avaliar novidade real, problemas de contexto em tarefas longas, custos elevados, fragilidade em validações complexas e riscos de segurança.

Estamos avançando rapidamente, mas ainda não chegamos à autonomia plena.

E talvez a pergunta mais importante não seja “quando a IA fará ciência sozinha?”, mas “quem será capaz de auditar ciência produzida com IA?”.

O erro das manchetes: emoção antes da precisão

Títulos como “Elon Musk chocou a OpenAI” ou “novo modelo destrói concorrentes” são produtos do ambiente algorítmico.

Eles existem porque plataformas de vídeo e redes sociais recompensam emoção, urgência e conflito.

O problema não é apenas o criador de conteúdo. É o incentivo do meio.

O algoritmo raramente premia a frase mais precisa. Ele premia a frase mais clicável.

Isso distorce a compreensão pública da tecnologia. Uma melhoria real vira revolução absoluta. Um benchmark específico vira prova de supremacia geral. Uma demonstração controlada vira previsão sobre o fim de uma profissão.

Para entender IA com seriedade, é preciso resistir à estética da guerra.

Nem todo lançamento é uma ruptura. Nem todo ranking muda o mercado. Nem toda promessa de autonomia se sustenta em produção.

A pergunta correta não é “quem chocou quem?”.

A pergunta correta é: o que foi comprovado, o que é inferência plausível e o que ainda é narrativa promocional?

Essa separação parece simples, mas é uma das habilidades mais importantes da era da IA.

Porque quanto mais impressionantes ficam os sistemas, mais fácil se torna confundir demonstração com realidade.

O impacto real sobre desenvolvedores

A consequência mais provável não é o desaparecimento imediato dos programadores.

A consequência mais realista é a mudança do que significa ser bom em programação.

O desenvolvedor que apenas executa tarefas simples e repetitivas tende a perder espaço. O profissional capaz de especificar problemas claramente, avaliar soluções, entender

arquitetura, revisar código, criar testes e orquestrar agentes torna-se muito mais valioso.

A habilidade central deixa de ser apenas escrever código. Passa a ser formular bem o trabalho.

Isso inclui saber pedir, dividir tarefas, montar contexto, validar respostas, identificar riscos, revisar decisões e impedir que o agente avance na direção errada.

Mas isso também cria uma divisão nova.

De um lado, estarão profissionais capazes de usar agentes como multiplicadores de capacidade. Do outro, profissionais que aceitarão respostas automáticas sem entender o que está sendo produzido.

Quanto mais poderosa a IA, mais perigosa ela se torna nas mãos de quem não sabe avaliar o resultado.

A IA acelera um bom engenheiro.

Mas também amplifica a confusão de quem não sabe o que está fazendo.

Esse é o paradoxo: a IA reduz a barreira para produzir software, mas aumenta a exigência para validar software.

O custo invisível da autonomia

Autonomia custa caro.

Um agente que trabalha por horas consome grandes volumes de tokens, executa dezenas ou centenas de ferramentas, gera múltiplas tentativas, cria logs, chama APIs, processa arquivos e exige infraestrutura pesada.

Se o custo por tarefa for alto demais, a autonomia não escala economicamente.

Empresas não adotarão agentes em massa apenas porque são impressionantes. Elas adotarão quando forem confiáveis, auditáveis e economicamente vantajosos.

A pergunta decisiva será: quanto custa transformar uma especificação em uma entrega funcional, testada e segura?

Se o agente economiza tempo, mas gera retrabalho, riscos ou custos imprevisíveis, seu valor diminui. Se entrega com consistência, documentação, testes e baixo custo, então muda completamente a economia do software.

Autonomia sem confiabilidade é apenas risco automatizado.

Essa talvez seja a frase que melhor resume o momento.

A indústria está tentando aumentar autonomia. Mas o mercado vai exigir confiabilidade.

E confiabilidade não nasce de demonstrações bonitas. Nasce de auditoria, testes, rastreabilidade, rollback, segurança e responsabilidade.

Privacidade, propriedade intelectual e concentração de poder

Dados de trabalho real são extremamente sensíveis. Eles podem revelar código proprietário, estratégias de produto, falhas de segurança, padrões internos, arquitetura, decisões técnicas e conhecimento acumulado.

Se ferramentas de IA coletam ou aprendem com esses fluxos, surge uma pergunta central: quem é dono desse aprendizado?

- O usuário?
- A empresa que escreveu o código?
- A plataforma que capturou o comportamento?
- O laboratório que treinou o modelo?

Essa questão ainda não está resolvida. E ela será cada vez mais importante.

Além disso, existe o risco real de concentração de poder. Se poucas empresas controlarem os modelos, as IDEs, os agentes, a infraestrutura de execução e os dados de workflow, elas não estarão apenas vendendo ferramentas.

Estarão intermediando o próprio trabalho intelectual de milhões de profissionais.

A IA para programação pode se tornar a nova camada de plataforma do software global.

Assim como sistemas operacionais, navegadores, lojas de aplicativos e nuvens concentraram poder em fases anteriores da computação, agentes de IA podem se tornar a próxima camada dominante.

Quem controla a camada onde o trabalho é executado controla muito mais do que uma ferramenta.

Controla o caminho entre intenção e produção.

O risco da dependência invisível

Existe ainda um risco menos discutido: a dependência cognitiva.

Quando uma equipe passa a usar agentes para planejar, escrever, revisar, testar e documentar, ela não terceiriza apenas tarefas. Ela começa a terceirizar parte do próprio raciocínio operacional.

No início, isso parece apenas ganho de produtividade.

Com o tempo, pode virar perda de autonomia.

Se uma empresa não entende mais como seu sistema foi alterado, apenas aceita os diffs gerados por agentes. Se desenvolvedores deixam de cultivar julgamento técnico porque o agente “resolve”. Se decisões de arquitetura passam a ser aceitas por conveniência. Então a organização não ganhou apenas velocidade. Ganhou também uma nova dependência.

A pergunta não é se devemos usar agentes.

Devemos.

A pergunta é: como usar agentes sem perder domínio sobre o próprio trabalho?

A resposta passa por documentação, revisão humana, testes, rastreabilidade, governança e cultura técnica.

A IA deve ampliar a inteligência da equipe, não substituir sua capacidade de julgamento.

Da era do modelo para a era do sistema

Estamos saindo da era do modelo como produto e entrando na era do sistema como produto.

O modelo ainda importa.

Mas sozinho não basta.

O diferencial passa a ser o conjunto completo: modelo, dados, interface, terminal, IDE, contexto, memória, ferramentas, execução, permissões, testes, segurança, observabilidade e integração com o fluxo real de trabalho.

Essa mudança lembra transições anteriores da tecnologia. Em certo momento, o processador era o centro da disputa. Depois, o sistema operacional virou a camada estratégica. Mais tarde, a nuvem e os ecossistemas dominaram.

Agora, a camada emergente pode ser o agente: o intermediário inteligente entre intenção humana e execução digital.

Quem controlar essa camada capturará valor imenso.

Não apenas por ter o melhor modelo.

Mas por estar dentro do fluxo de trabalho.

Nem apocalipse, nem marketing ingênuo

A chegada dos agentes de IA à programação não deve ser tratada nem como apocalipse imediato nem como propaganda vazia.

Ela é uma transformação real, mas ainda incompleta.

Os agentes estão ficando melhores em planejar, codificar, testar e operar ferramentas. Sistemas como Grok Build, Qwen e outros agentes avançados mostram que a indústria está

migrando rapidamente para workflows mais autônomos.

Mas os limites continuam claros: confiabilidade, custo, privacidade, segurança, avaliação de novidade, reprodutibilidade, concentração de poder e controle humano.

A pergunta mais importante não é se a IA vai substituir todos os desenvolvedores.

A pergunta é quais desenvolvedores, empresas e países saberão usar agentes como multiplicadores de capacidade sem entregar a eles todo o controle operacional.

A vantagem não estará apenas em ter acesso à melhor IA.

Estará em saber orquestrá-la.

O futuro próximo do software não será escrito apenas por humanos nem apenas por máquinas. Será construído por equipes híbridas - mas essa frase só é verdadeira se os humanos continuarem capazes de compreender, auditar e contestar o que os agentes produzem.

Caso contrário, “equipe híbrida” vira apenas uma forma elegante de dizer dependência automatizada.

A corrida, portanto, não é só por inteligência artificial.

É por controle da nova infraestrutura do trabalho intelectual.

E nessa nova fase, vencer não será apenas ter o modelo mais poderoso.

Será construir o sistema onde o trabalho acontece - sem esquecer que todo sistema poderoso também precisa de limites, fiscalização e responsabilidade.

Referências contextuais

Estas referências foram usadas como base factual para orientar o contexto do artigo. O texto principal é opinativo e analítico, não uma reprodução das fontes.

xAI - Grok Build CLI
https://x.ai/news/grok-build-cli

Qwen - Qwen3.7: The Agent Frontier
https://qwen.ai/blog?id=qwen3.7

From Copilots to Colleagues: A Survey of Autonomous Research Agents
https://victorchen96.github.io/auto_research_survey.pdf